**Reddit Dating Behavior Analysis**

**1. Introduction**

**1.1 Overview**

Our project analyzes behavioral patterns within the Reddit dating community, aiming to identify common traits, engagement levels, and communication styles among users. It also aims to develop a recommendation system that can suggest compatible user profiles based on their preferences.

**1.2 Objectives**

* **Data Processing and Analysis**: Collect and preprocess Reddit data from specific dating subreddits.
* **User Behavior Insights**: Develop behavioral metrics to understand user interactions.
* **Recommendation System**: Build a system to match users based on behavior and preferences.
* **Visual Analytics**: Provide in-depth visualizations for analyzing user metrics.

**2. Directory Structure**

The directory structure for my project organizes configuration files, data storage, analysis scripts, and visualization output:

|  |
| --- |
| reddit\_dating\_analysis/  ├── config/  │ └── credentials.py # Stores Reddit API credentials  ├── data/  │ ├── raw/ # Raw data files from Reddit API  │ ├── processed/ # Preprocessed data files  │ └── visualizations/ # Saved visualizations (HTML format)  ├── analysis/  │ ├── data\_processing.py # Data processing functions  │ ├── behavior\_analyzer.py # Analysis of user behavior patterns  │ ├── recommendation\_system.py # Recommendation model based on user preferences  │ └── visualizer.py # Visualization and dashboard creation  └── data\_collection.py # Scrape dataset from reddit API  └── main.py # Main script to run the analysis and visualizations |

Each directory and file serves a specific purpose, from storing data and credentials to executing data analysis and generating visualizations.

**3. Data Collection**

**3.1 Reddit Data Collection**

I collected data from Reddit using the PRAW library (Python Reddit API Wrapper), which enables access to specific subreddit posts and comments. Key details about each post and its comments are captured, including:

* **Post Attributes**: Unique post ID, title, body text, author, timestamp, and score.
* **Comment Attributes**: Comment ID, author, body text, score, and timestamp.

The data\_collection.py script, located in config/, handles authentication with the Reddit API using credentials stored in credentials.py. This script retrieves posts and their comments, limiting data to recent interactions from the past three months. The collected data is saved in JSON format for further processing.

**4. Data Processing Pipeline**

**4.1 Preprocessing**

The preprocessing script, data\_processing.py, carries out several steps to clean and structure the data:

* **Text Cleaning**: Removes unwanted text elements like URLs, punctuation, and stop words. The script uses NLTK's stop words list and tokenizes text to filter noise.
* **Feature Extraction**: Extracts both temporal and textual features:
  + **Time Features**: Converts the timestamp to indicate hour, day of the week, and time of day (categorized as morning, afternoon, evening, or night).
  + **Textual Features**: Calculates metrics such as word count, sentence count, and average word length. Sentiment analysis is performed using TextBlob to evaluate positive, neutral, or negative sentiment.

**4.2 Comment Processing**

Comment features are extracted through detailed analysis, assessing:

* **Sentiment**: Calculated for each comment using TextBlob, providing a polarity score.
* **Response Times**: Measures time intervals between user responses to estimate typical response behavior.
* **Engagement Levels**: This feature estimates user activity and responsiveness, categorizing engagement as low, medium, or high based on the comment count and average response time.

**5. Models and Analysis**

**5.1 Recommendation System**

The recommendation\_system.py script implements a recommendation model using the K-Nearest Neighbors (KNN) algorithm. Our model identifies compatible user profiles based on selected behavioral features, including:

* **Communication Style**: Matches users based on sentiment preferences (positive, neutral, or critical).
* **Response Time**: Finds users who prefer similar response speeds.
* **Engagement Level**: Matches users with similar interaction levels.

**Algorithm Details**

The recommendation system uses the KNN model with cosine similarity to evaluate user compatibility. For this, the feature vectors are standardized:

* **Behavioral Features**: These include average response time, total comments, sentiment mean, and average word count per comment. The features are scaled using StandardScaler.
* **Textual Features**: The TF-IDF vectorizer converts user text into numerical values, enabling analysis of users’ language use and communication styles.

**5.2 Model Training**

Feature vectors are prepared by combining behavioral metrics with sentiment scores. KNN’s similarity scoring calculates closeness based on user preferences:

* **Scoring**: Each match is ranked based on sentiment alignment, response time proximity, and engagement similarity. The scoring system weights factors based on user priority.

**6. Visualizations**

**6.1 Behavior Dashboard**

An interactive dashboard is generated using Plotly, with subplots for multiple user metrics. This includes:

* **Response Time Distribution**: Histogram showing the range of user response times.
* **Engagement Level**: Pie chart visualizing engagement categories.
* **Communication Styles**: Pie chart indicating sentiment distributions.
* **Message Frequency vs. Response Time**: Scatter plot showing how frequently users post versus their typical response time.

The dashboard enables interactive exploration of user behavior, providing insights into how different factors correlate with engagement and communication style.

**6.2 User Comparison Visualization**

This visualization allows for side-by-side comparisons of two users based on behavior metrics. Key differences in response times, frequency, engagement levels, and sentiment are highlighted, assisting users in identifying common ground or potential challenges in communication.

**7. Results and Insights**

**7.1 Behavioral Patterns**

* **Response Time Trends**: Users typically respond within 1 to 12 hours, indicating moderate engagement.
* **Engagement Distribution**: Most users exhibit medium engagement, with regular but not excessive posting and interaction.
* **Communication Style**: Users tend toward neutral or positive sentiment, with fewer displaying negative tendencies.

**7.2 Recommendation Effectiveness**

The recommendation system provides accurate matches based on user preferences, such as sentiment and engagement levels, making it a useful tool for suggesting compatible profiles.

**8. Challenges and Limitations**

**8.1 Data Limitations**

* **Reddit API Constraints**: Data collection limitations restrict the scope of available posts and comments.
* **Sentiment Analysis Constraints**: Basic sentiment scoring from TextBlob might lack the depth to capture nuanced emotions, particularly in complex discussions.

**8.2 Model Limitations**

* **KNN Scalability**: KNN may slow down with large datasets, and methods such as dimensionality reduction could improve performance in future iterations.

**9. Conclusion and Future Work**

This project provides valuable insights into dating-related behavior on Reddit. Future directions could include:

* **Advanced Sentiment Analysis**: Integrating deep-learning-based sentiment analysis for more nuanced emotional detection.

**10. Appendix**

**10.1 Setup**

* **Reddit API**: API credentials are saved securely in config/credentials.py.